Programación I

Curso 2015/2016

Apellidos *(rellenar con letra clara)*:	Nombre:
Grupo: DNI:	

● El tiempo para realizar este examen es de **2 horas**.

Eval2

14/01/2016

Ejercicio 1

Se necesita desarrollar una aplicación que gestione una estación de Inspección Técnica de Vehículos (ITV). Cada inspección incorpora los siguientes datos: la matrícula del vehículo (un String), la categoría del vehículo (un número entero en el rango [1, 3] que representará el tipo de vehículo: 1 para turismo, 2 para furgoneta y 3 para camión), tres valores de tipo boolean que representarán respectivamente si el vehículo ha superado o no las pruebas de luces, dirección y frenos (un valor *true* significará que se ha pasado la prueba y un valor *false* que no) y por último el nivel de emisión de gases como un número real (double).

Define en Java un tipo compuesto/clase *Inspección* (y su correspondiente constructor) que represente una inspección de la ITV. **(0,5 puntos)**

```
static class Inspeccion {
  String matricula;
  int categoria;
  boolean luces;
  boolean direccion;
  boolean frenos;
  double gases;
}
static Inspeccion consInspeccion
  (String m, int cat, boolean l, boolean dir, boolean fren, double gas) {
  Inspeccion resultado = new Inspeccion();
  resultado.matricula = m;
  resultado.categoria = cat;
  resultado.luces = l;
  resultado.direccion = dir;
  resultado.frenos = fren;
  resultado.gases = gas;
  return resultado;
}
```

Define en Java cuatro datos de tipo *Inspección* y de nombres *turismo0*, *turismo1*, *camion* y *furgoneta* que representen respectivamente las siguientes inspecciones:

- *turismo0*: Una inspección pasada a un turismo de matrícula 1003-JHK que no ha pasado ninguna de las pruebas de luces, dirección y frenos y que ha dado nivel 0 en la de emisión de gases.
- *turismo1*: Una inspección pasada a un turismo de matrícula 5003-POI que ha pasado las tres pruebas de luces, dirección y frenos y que ha dado 0.5 en la de emisión de gases.
- *camion*: Una inspección pasada a un camión de matrícula 3924-JHT que ha pasado las pruebas de luces y dirección pero no así la de frenos y que ha dado 1.5 en la de emisión de gases.
- *furgoneta*: Una inspección pasada a una furgoneta de matrícula 6579-TSS que ha pasado las pruebas de luces y de frenos, pero no ha pasado la de dirección y que ha dado 2.5 en la de emisión de gases.

(0,5 puntos)

```
Inspeccion turismo0 =
  consInspeccion("1003-JHK", 1, false, false, false, 0.0);
Inspeccion turismo1 =
  consInspeccion ("5003-POI", 1, true, true, true, 0.5);
Inspeccion camion =
  consInspeccion ("3924-JHT", 3, true, true, false, 1.5);
Inspeccion furgoneta =
  consInspeccion ("6579-TSS", 2, true, false, true, 2.5);
```

Define en Java un array que esté vacío (*itv0*), otro formado por las inspecciones *turismo0* y *turismo1* definidas anteriormente (*itv2*) y otro que represente una colección de inspecciones que contenga las cuatro que definiste previamente en el orden dado (*itv4*). **(0,5 puntos)**

```
Inspeccion[] itv0 = {};
Inspeccion[] itv2 = {turismo0, turismo1};
Inspeccion[] itv4 = {turismo0, turismo1, camion, furgoneta};
```

Define en Java un visualizador para un array de inspecciones. Se supone que ya está definida la operación *verInspeccion*, que convierte a String una inspección. **(1 punto)**

```
static String verInspecciones (Inspeccion [] inspecciones) {
  String resultado = "[";
  for (int i = 0; i < inspecciones.length; i ++) {
    if (i != 0) {
     resultado = resultado + "," + "\n";
    }
    resultado = resultado + verInspeccion(inspecciones[i]);
  }
  return resultado + "]";
}
```

Ejercicio 2

Se considera que una inspección es pasada si el vehículo sometido a la inspección ha superado las pruebas de luces, dirección y frenos y su nivel de emisión de gases es menor ($<$) que 2.0. Define en Java la función *pasaLaInspeccion*, que determine si una inspección ha sido pasada o no, es decir, cuyo resultado es *true* si la inspección ha sido pasada y *false* en caso contrario. **(0,5 puntos)**

```
static boolean pasaLaInspeccion (Inspeccion inspeccion)
{
  return inspeccion.luces && inspeccion.direccion &&
         inspeccion.frenos && (inspeccion.gases < 2.0);
}
```

Ejercicio 3

Se ha fijado la tasa a pagar por una inspección (en euros) como 50 para turismos, 70 para furgonetas y 120 para camiones. Define en Java la función *tasa*, que calcule la tasa a pagar por una inspección. **(0,5 puntos)**

```
static double tasa (Inspeccion inspeccion)
{
  if (inspeccion.categoria == 1)
    return 50;
  else if (inspeccion.categoria == 2)
    return 70;
  else
    return 120;
}
```

Se quiere definir una función *recaudacion* que a partir de una colección de inspecciones representada por un array, calcule el total recaudado en euros por la estación.

Indica qué devolverá la función *recaudacion* si recibe los siguientes datos:

1. El array vacío *itv0* definido anteriormente.
2. El array formado por los vehículos *turismo0* y *turismo1* definidos anteriormente (*itv2*).
3. El array *itv4* definido anteriormente.

(0,5 puntos)

```
recaudacion(itv0) = 0
recaudacion(itv2) = 100
recaudacion(itv4) = 290
```

Define la función *recaudacion* en Java. **(1,5 puntos)**

```
static double recaudacion (Inspeccion [] inspecciones)
{
  double total = 0;
  for (int i = 0; i < inspecciones.length; i++)
    total = total + tasa(inspecciones[i]);
  return total;
}
```

Ejercicio 4

Se quiere definir una función *tasaMatricula* que a partir de una matricula de un vehículo y una colección de inspecciones, devuelva la tasa a pagar por dicho vehículo en euros. En caso de no encontrarse el vehículo en la colección, se devolverá el valor -1 como resultado.

Define la función *tasaMatricula* en Java. **(2 puntos)**

```
/**
* FUNCION tasaMatricula (String matricula, Inspeccion[] inspecciones) --> Real
* PRE: cierto
* POST: Devuelve la tasa a pagar por el vehículo de ''inspecciones'' que tiene
*       esta "matricula".
*/
static double tasaMatricula (String matricula,
                             Inspeccion[] inspecciones) {
  boolean encontrado = false;
  double resultado = -1;
  int i = 0;
  while (i < inspecciones.length && !encontrado) {
    if (inspecciones[i].matricula.equals(matricula)) {
      resultado = tasa(inspecciones[i]);
      encontrado = true;
    }
    else {
      i = i + 1;
    }
  }
  return resultado;
}
```

Ejercicio 5

Se quiere definir una función *masContaminante* que a partir de una colección de inspecciones devuelva la matrícula del vehículo más contaminante entre todas ellas. Se supone que la colección tiene al menos un elemento. Indica qué devolverá la función *masContaminante* si recibe los siguientes datos:

1. El array *itv2* definido arriba.
2. El array *itv4* definido arriba.

(0,5 puntos)

```
masContaminante(itv2) = "5003-POI"
masContaminante(itv4) = "6579-TSS"
```

Define la función *masContaminante* en Java. **(2 puntos)**

```
/**
* FUNCION masContaminante (Inspeccion[] inspecciones)  ---> String
* PRE: inspecciones.length > 0 (''inspecciones'' tiene al menos un elemento)
* POST: ''resultado'' es la matrícula del vehiculo más contaminante entre todas
*       "inspecciones".
*/
static String masContaminante (Inspeccion [] inspecciones) {
  String resultado = "";
  double mayor = 0;
  for (int i = 0; i < inspecciones.length; i++) {
    if (inspecciones[i].gases > mayor) {
      resultado = inspecciones[i].matricula;
      mayor = inspecciones[i].gases;
    }
  }
  return resultado;
}
```
